Machine Learning Modeling to Predict Rain

Springboard Data Science Capstone Project

By Sangeeta Jayakar, PhD - July 14, 2021

Introduction

Weather predictions have long been an important aspect of human civilization. It can impact so many things such as agriculture, travel, real estate, and overall quality of life. Before modern science, people would attempt to forecast the rain by simply observing the clouds. The development of tools to measure atmospheric properties such as humidity, pressure, and temperature led to greater advances in the abilities of weather predictions. This has been enhanced by the utilization of weather data collected all over the globe. Weather scientists use these data to help make their daily weather forecasts as well as predicting extreme weather occurrences. Still, weather predications are not always 100% accurate. For this project, I would like to determine if machine learning models can be used to predict the weather.

Problem Statement:

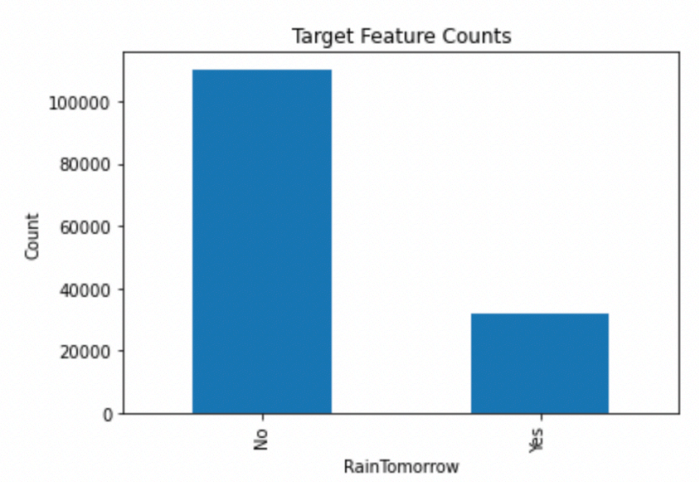
Can machine modeling of weather data be used to accurately predict if it will rain?

Data Wrangling and Cleaning

The dataset used for this project was found on kaggle.com. It contains various weather observations collected daily from Australia's Bureau of Meteorology over a period of 10 years from 49 locations across Australia. These weather observations totaled 145,460 for 23 weather-related features:

['Date', 'Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustDir', 'WindGustSpeed', 'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am', 'Temp3pm', 'RainToday', 'RainTomorrow']

The target feature is the last column labeled "RainTomorrow" which contains either a Yes or No entry for whether or not it rained (more than 1mm) the following day. There were 3267 rows that did not have an entry for RainTomorrow. These rows were dropped. The 'Yes' and 'No' in the RainTomorrow columns were replaced with '1' and '0' so that these numeric values could be used for exploratory data analysis such as for finding the ratio of Rain/NoRain. This same replacement was done for the RainToday column.



After exploration of the target feature, some new columns were created. For example, there was a column called "Pressure9am" and "Pressure3pm" which recorded the air pressure at those two times each day. I created a new column called "PressureDiff" which was the difference between the "Pressure3pm" and "Pressure9am" in order to explore whether a big change in air pressure during the day was actually more helpful in predicting next day rain. I also explored the relationship between the "9am" and "3pm" columns for other weather features ('WindDir', 'WindSpeed', 'Humidity', 'Pressure', 'Cloud', and 'Temp') and concluded that the '9am' and '3pm' columns for each feature was redundant. Ultimately all of the '9am' columns

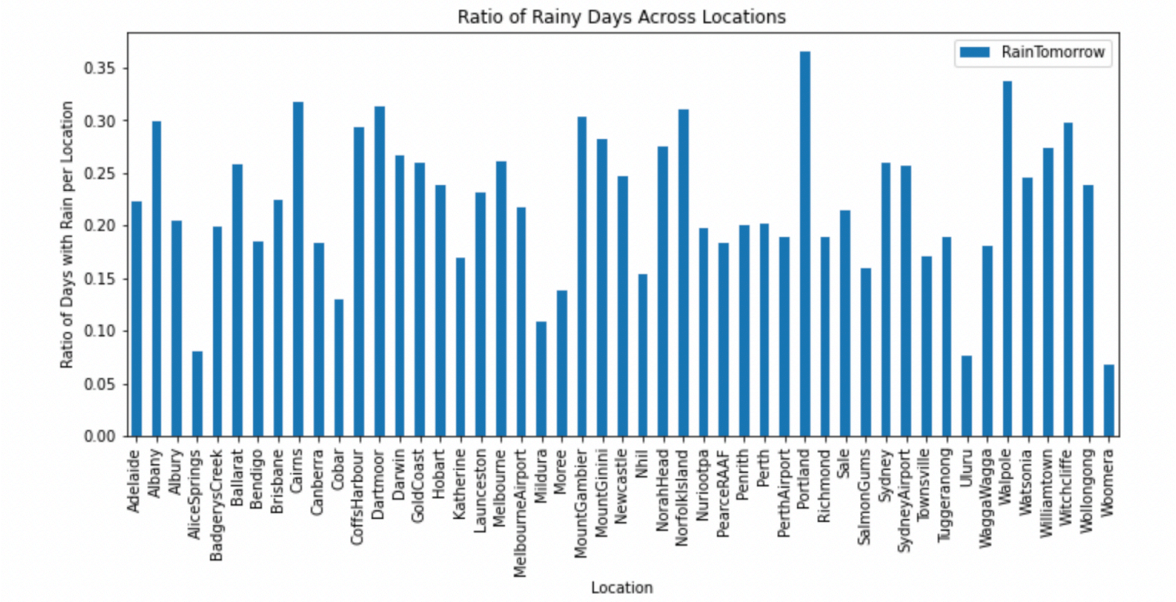
were dropped so that only the '3pm' and the 'Diff' columns would be used for the modeling.

Lastly, I extracted the month and year out of the date column so that observations could be grouped by month, to take into account different seasonal weather patterns.

Exploratory Data Analysis

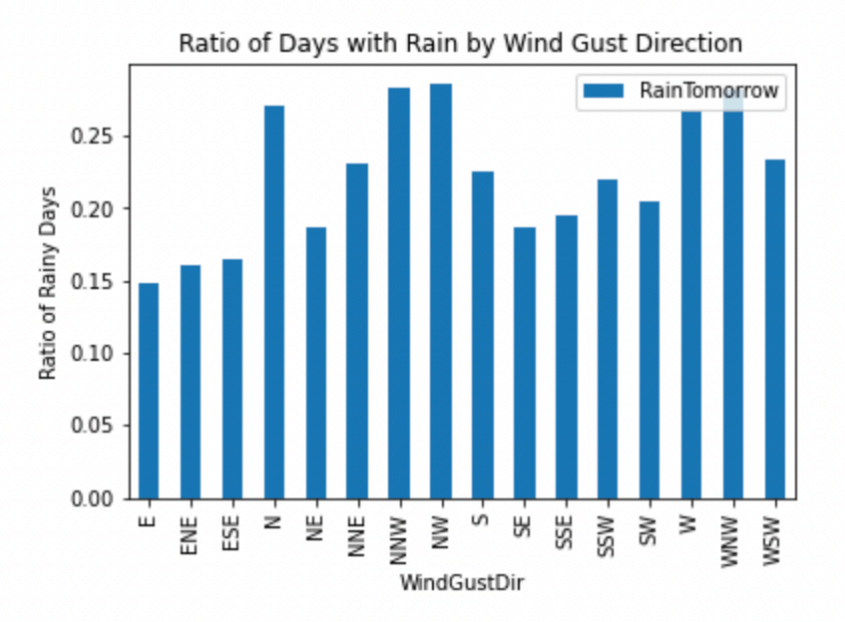
Looking at the 'RainTomorrow' column, the mean of the column could be calculated and that would give the ratio of days with rain (because 'Yes' =1 and 'No' =0). This strategy would be used throughout the rest of the EDA section for the various features being analyzed. By looking at the counts of number of 'Yes' and 'No' entries, it is apparent that there are about three times more 'No's, so these are unbalanced.

The categorical features were explored first in the EDA phase of this study. The location column contained 49 unique locations, each with a similar number of observations around 3,000 except for 3 locations which had about half. Looking at the ratio of days with rain grouped by location suggested that location would be an important feature in the model (chi-square = 3544, $p<0.001$)

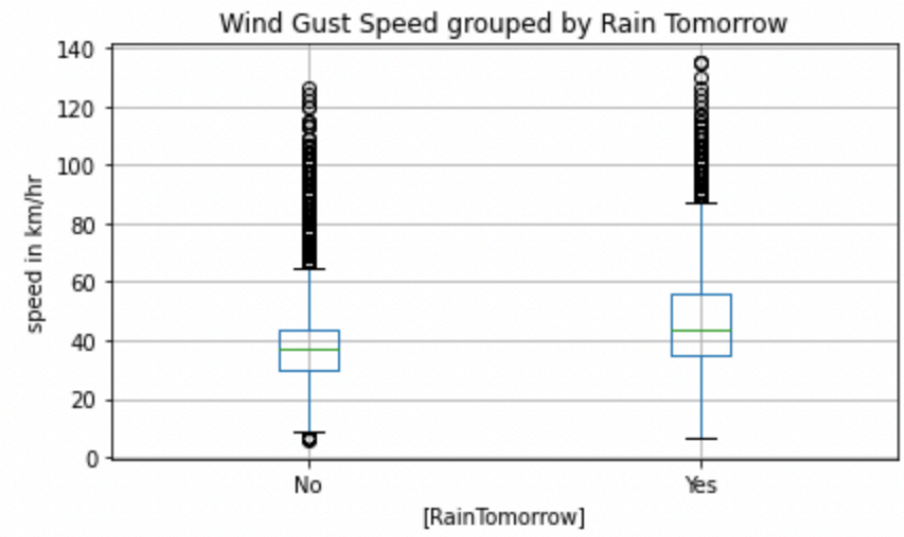


Another categorical feature was the wind gust direction. It also looked like the direction of the strongest daily wind gust would have an impact on the rain forecast.

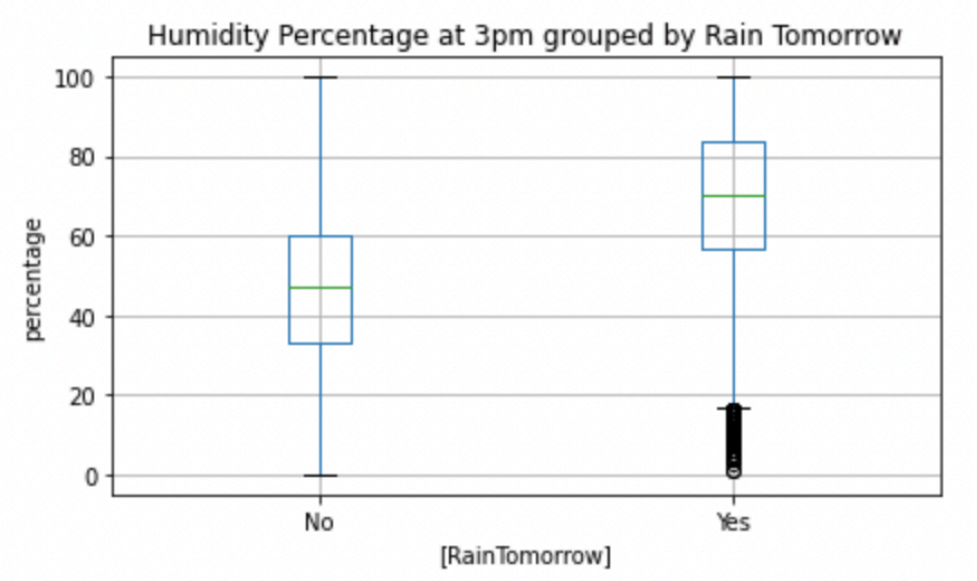
From this graph, it looks like when the strongest winds come from the north or northwest, there is more likely to be rain (chi-square=1519, p<0.001).



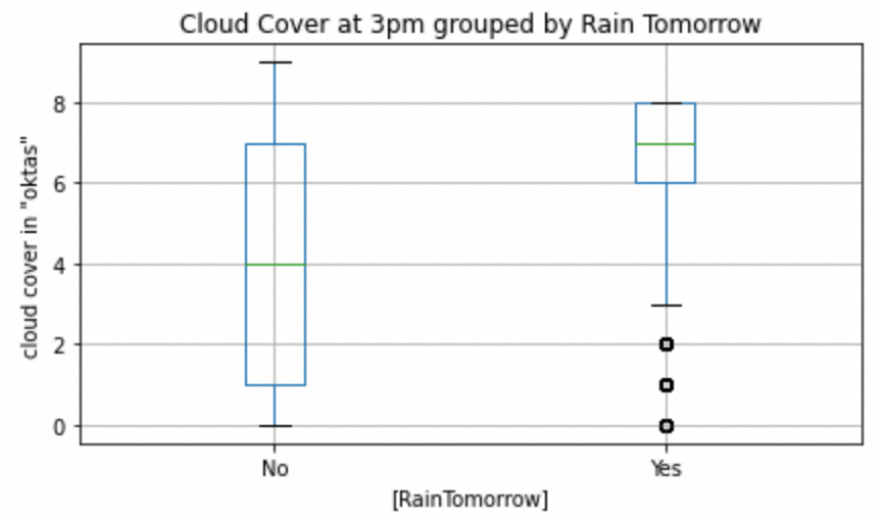
Most of the columns in the dataset contained numerical data. The 'WindGustSpeed' column contained the speed of the strongest wind gust in km/hour recorded for that particular day. This value was higher when there was rain the next day (t-test, p<0.001).



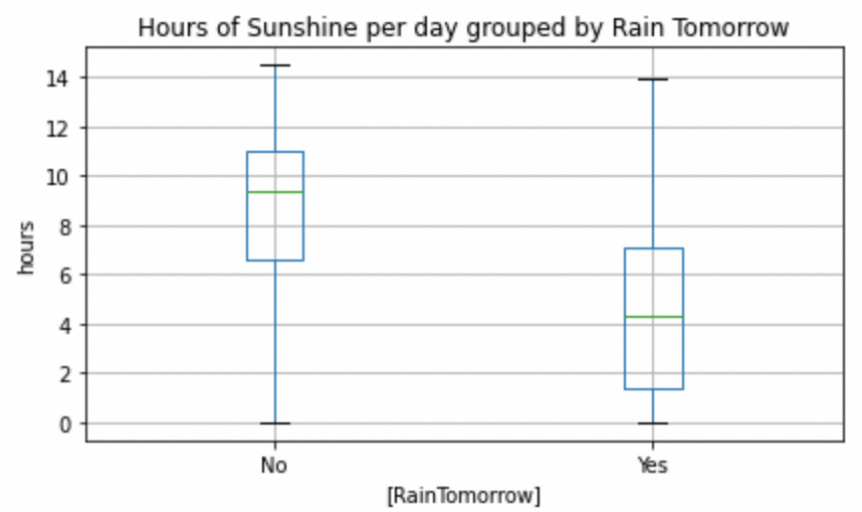
The ‘Humidity3pm’ measurement was found to be significantly higher when there was recorded next-day rain (t-test, $p<0.001$).



The ‘Cloud3pm’ column recorded the fraction of the sky covered by cloud measured in ‘oktas’, or units of eighths. A measurement of 0 indicates no cloud cover, while a measurement of 8 indicates the entire sky is covered. When grouping the data by the RainTomorrow column, the ‘Cloud3pm’ has a median value of 4 for no next day rain, while it has a median value of 7 when there is next day rain. By t-test, the cloud cover at 3pm is significantly greater when there is next day rain ($p<0.001$).

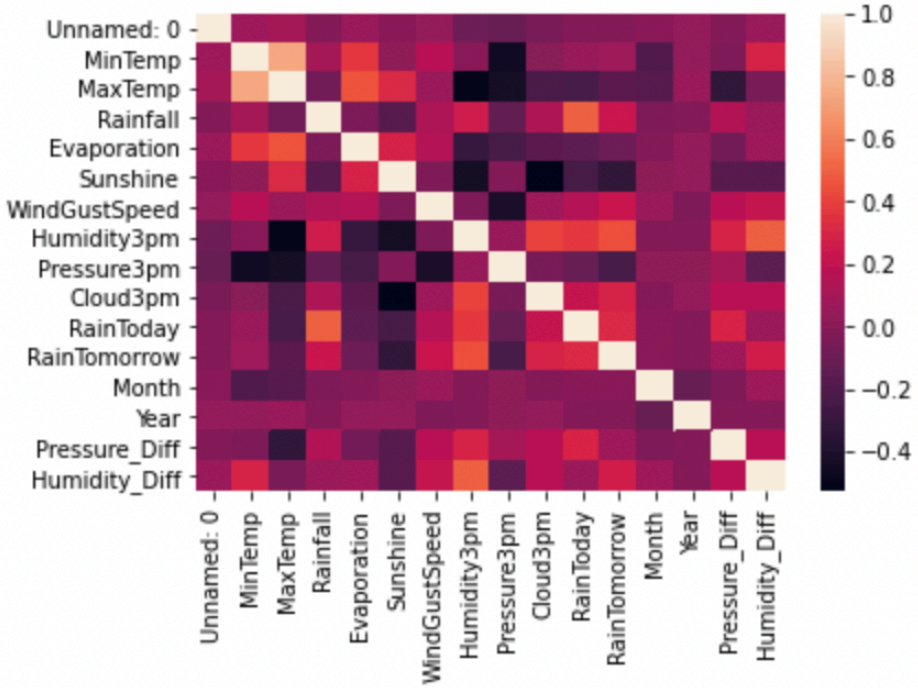


Another feature explored was the 'Sunshine' column which depicts the recorded number of hours of sunshine per day. As one might imagine, this followed the opposite trend of the 'Cloud3pm' column. More cloud cover is directly related to less sunshine. This trend can be seen in the boxplot below where the median hours of sunshine was about 9 hours when there was no rain the next day compared to about 4 hours of



sunshine when there was rain the next day.

The correlation of all of the numerical features was calculated and used to make a heatmap. Looking across at the RainTomorrow row, there are some features that show a correlation as indicated by the color contrast. Of note, the 'Humidity3pm' column forms



a lighter orange color on the color scale while the 'Sunshine' column forms a noticeably darker color, indicating that these features are positively and negatively correlated with 'RainTomorrow', respectively.

Pre-processing

Following the exploratory data analysis, columns that were determined to be unnecessary were dropped, so that the final columns that remained, not including the target feature 'RainTomorrow' were:

['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustDir', 'WindGustSpeed', 'Humidity3pm', 'Pressure3pm', 'Cloud3pm', 'RainToday', 'Month', 'Pressure_Diff', 'Humidity_Diff']

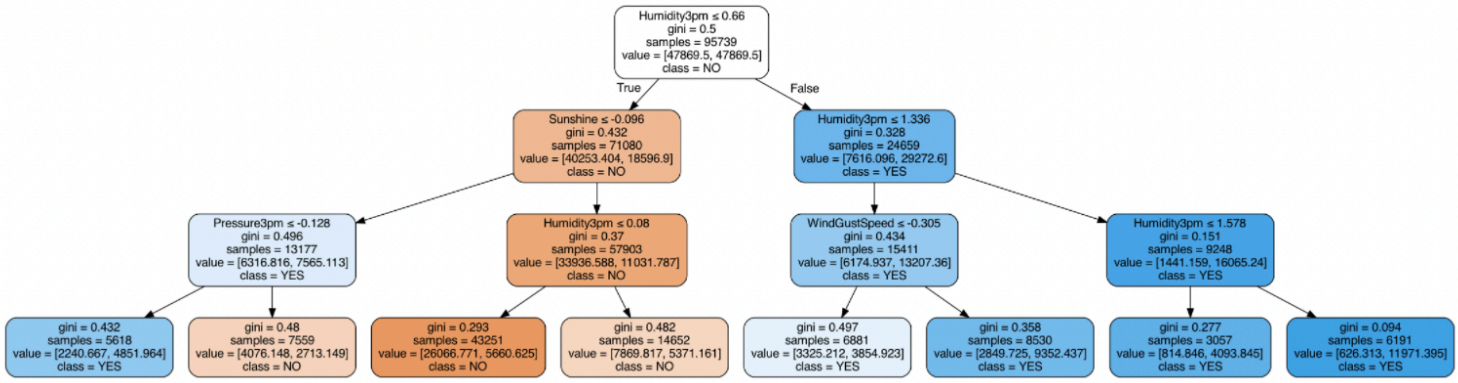
Dummy variables were created for categorical features. The 'Month' column was converted to a string datatype with names of months so that this column could be categorical, allowing dummy variables to be made out of this information. Finally, the data were scaled prior to modeling using Scikit-learn's StandardScaler function.

Modeling

I chose to evaluate four supervised machine learning models that can be used for classification problems: decision tree, random forest, gradient boosting, and logistic regression. To evaluate the performance of the models, I had to first choose a performance metric. I decided to use the F1 score. The F1 score is defined as a 'harmonic mean between Precision and Recall' and is useful when you have a small unbalanced positive class, which we do, since the the RainTomorrow class is highly unbalanced.

Decision Tree

The first model that was tested was a Decision Tree. A preliminary tree was generated using the Gini impurity criteria with a max depth of of 3. This was examined with and without class weight balancing. A higher F1 score was yielded with class weight balancing (F1 score = 0.586) compared to without (F1 score = 0.476). Also, we know from the counts for 'Yes' and 'No' that this class is highly unbalanced. There are

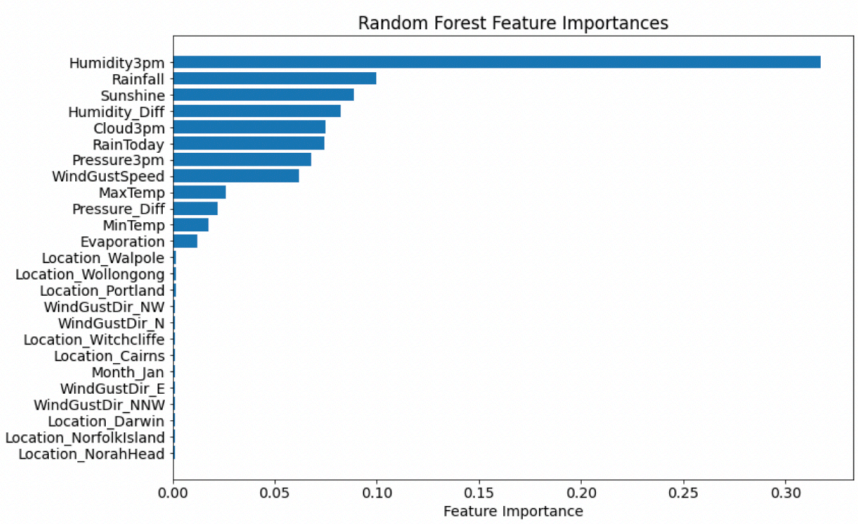


approximately three times as many 'No's compared to 'Yes's. Looking at the visual of the decision tree made with Graphviz, you can see some features that are prominent in the tree as being important in the classification: 'Humidity3pm', 'Sunshine', 'WindGustSpeed', and 'Pressure3pm'. A function to test different max depths was used and found that the optimal max depth was 6. When the tree criteria was changed to a max depth of 6, an F1 score of 0.61 was yielded.

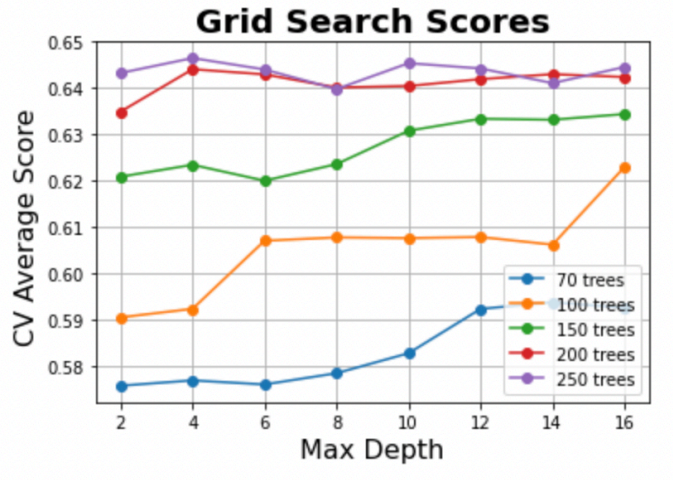
Decision trees are useful because they are generally easy to explain and visualize (see tree above). They can be tuned by adjusting the max depth, however, increasing the max depth too much can lead to overfitting. With decision trees, you run the risk of overfitting your tree and they can be less accurate than other machine learning algorithms. When there are many class labels, the calculations can become complex. Trees can be biased towards features with greater number of categories.

Random Forest

In order to prevent the overfitting resulting from Decision Tree models, I tried a Random Forest model. I plotted the feature importances, only showing the top 25



features. Interestingly, the top feature as ranked by importance is 'Humidity3pm', which is in agreement with what was seen in the decision tree. The next important features to note are 'Rainfall', 'Sunshine', and 'Humidity_Diff'. Also, two features that were seen on the decision tree, 'Cloud3pm' and 'WindGustSpeed' are also shown to be in the top 8 features.



I used GridSearchCV to find the best hyperparameters for max depth and number of trees. From plotting the GridSearchCV results and using the F1 score as the scoring function, I chose a max depth of 10 and 150 trees for the hyperparameters. Running the random forest with the chosen hyperparameters, resulted in an F1 score of 0.642.

Since random forest is made up of multiple decision trees, too large a number number of trees can make it a slower algorithm. However, it is much less prone to overfitting than the decision tree. Also, random forest can output the feature importances.

Gradient Boosting

I also evaluated another tree-based model with Gradient Boosting. The optimal hyperparameters were determined by choosing a range of values for learning rate from 0.05 to 1. Additionally the number of trees and max depth were also evaluated. I found that 200 was the optimal number of trees, the best max depth was 6, and the optimal learning rate was 0.1. This resulted in an F1 score of 0.626.

Unlike with random forest, with gradient boosting there is sequential improvement on the model with each tree. Thus, models tend to be more accurate, but

also have a higher chance of overfitting which can be overcome by applying L1 and L2 regularization. Feature importances can also be outputted with gradient boosting.

Logistic Regression

The last model I tried was logistic regression. I used GridSearchCV to test different values for the 'C' parameter, ranging from 0.001 to 100. The optimal value of 'C' was found to be 10, resulting in an F1 score of 0.623.

Logistic regression can be a good performing model for binary classification when the dataset is linearly separable. It is less prone to over-fitting when using L1 and L2 regularization and it does not require scaling of data. Performance is enhanced with feature engineering to remove unrelated features. It cannot be used to solve non-linear problems, so in these instances a decision tree is more useful. Since our rain dataset is very complex, this is not the best choice of model.

Model Name	F1 Score	Train Time	Best Hyperparameter Values
Decision Tree	0.611	0:00:00.841381'	{'max_depth': 6}
Random Forest	0.642	0:00:08.737272'	{'n_estimators': 150, 'max_depth': 10}
Gradient Boost	0.655	0:01:08.342800'	{'n_estimators': 200, 'max_depth': 6, 'learning_rate': 0.25}
Logistic Regression	0.620	0:00:00.852280'	{'C': 10}

In addition to the F1 score to measure model performance, I also used a date time function to record how long it took to train each of the models and generate a performance score. Those scores are listed in the table. What is most striking from these

measurements is how much longer the gradient boosting model took to run compared to the other models. This could be a problem when dealing with large amounts of data.

Discussion

After examining the four models for this project, I thought of areas that could be improved. I was shocked that 'location' was not found to be an important feature when common knowledge tells us that location and climate are related. The continent of Australia is very large and contains different types of climates. There is the famous 'outback' which is like a dessert, as well as regions that are tropical, and also temperate. If I would improve upon my dataset, I would create a new column and after looking up the 49 unique locations, I would classify each location to one of the main types of climate. This may lead to certain climates being more of a predictive feature of rain tomorrow. A similar approach could have been taken with the months and replacing them with seasons, since there may be certain seasons where rain is more likely.

Another change I could make is using a different method to impute missing values. I imputed with the median but I could have used another value such as the mean. Or I could have deleted more of these rows with missing values instead of imputing. Perhaps deleting the rows with missing values would have resulted in higher performance scores.

Conclusion

The Random Forest model was best model out of the four models tested in terms of a high performance score and shorter training time. All four models were fairly close in their performance, with F1 scores ranging from 0.611 to 0.655. The gradient boosting model had a slightly higher F1 score than random forest, but took much longer to train the model. The random forest model was much more efficient in terms of a shorter training time and a high performance score, making it the preferred model for this project. Logistic regression is probably not the best model to use because the data is not linear.

References:

The dataset:
Young, J. Rain in Australia, Version 2. Retrieved [March 2021] from [https://www.kaggle.com/jsphyg/weather-dataset-rattle-package]

Other sources:
Cahir, J. 'Weather Forecasting', brittannica.com <https://www.britannica.com/science/weather-forecasting/History-of-weather-forecasting>[accessed July 2021]

'The birth of the weather forecast.' bbc news.com, <https://www.bbc.com/news/magazine-32483678>[accessed July 2021]

'Climate of the world: Australia', <https://www.weatheronline.co.uk/reports/climate/Australia.htm>[accessed July 2021]

Swalin, Alira,'Choosing the Right Metric for Evaluating Machine Learning Models - part 2', <https://www.kdnuggets.com/2018/06/right-metric-evaluating-machine-learning-models-2.html> [accessed July 2021]

Mwiti, D., 'Gradient Boosted Decision Trees - A Conceptual Explanation', <https://www.kdnuggets.com/2021/04/gradient-boosted-trees-conceptual-explanation.html>[accessed July 2021]

Kumar, N. 'Advantages and Disadvantages of Logistic Regression in Machine Learning', <http://theprofessionalspoint.blogspot.com/2019/03/advantages-and-disadvantages-of.html>[accessed July 2021]